ESTIMATION SYSTEM USING FUNCTION POINT

HIMALA DEWI A/P JAGANATHAN

BACHELOR OF COMPUTER SCIENCE
(SOFTWARE ENGINEERING)

UNIVERSITI MALAYSIA PAHANG

ESTIMATION SYSTEM USING FUNCTION POINT

HIMALA DEWI A/P JAGANATHAN

This thesis is submitted in partial fulfilment of the requirements for the award of degree of Bachelor of Computer Science (Software Engineering)

Faculty of Computer Systems & Software Engineering

Universiti Malaysia Pahang (UMP)

ABSTRACT

Software cost estimation is one of the most important parts in the project planning phase to ensure that the project will lead to success. Function point is one of the best methods used in estimating the software cost and size. Function point focuses more on measure the functionality thus make its estimation accurate and efficient. Function points features such as independent of programming language, product design and documented method makes it as an advantage. Besides measure the size and cost, function point also helps to measure the estimating the effort, schedule and defect in the project. The prototype of this research was developed using Microsoft Visual Studio 2008.

ABSTRAK

Penganggaran kos Perisian adalah salah satu bahagian yang paling penting dalam fasa perancangan projek untuk memastikan projek itu akan membawa kepada kejayaan. Fungsi titik adalah salah satu kaedah terbaik yang digunakan dalam menganggar kos perisian dan saiz. Titik fungsi lebih tertumpu kepada pengukur fungsi itu membuat anggaran yang tepat dan cekap. Titik fungsi mempunyai ciri-ciri seperti bebas daripada bahasa pengaturcaraan, reka bentuk produk dan kaedah didokumenkan menjadikan ia sebagai satu kelebihan. Selain mengukur saiz dan kos, titik fungsi juga membantu untuk mengukur menganggarkan usaha, jadual dan kecacatan dalam projek itu. Prototaip kajian ini telah dibina dalam Microsoft Visual Studio 2008

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

FP: Function Point

LOC: Lines of Codes

SLOC: Source Lines of Codes

COCOMO: Constructive Cost model

FPA: Function Point Analysis

IFPUG: International Function Point Users Group

ISBSG: International Software Benchmarking Standards Groups

ISO : International Organization for Standardization

CFP : Crude Function Point

RCAF: Relative Complexity Adjustment Factor

DFD : Data Flow Diagram

SRS: Software Requirement Specification

CHAPTER 1

1.1 Introduction

Software cost estimation is one of the most important parts in the software planning phase. A good planning and requirements from the beginning will lead a project to success. In software projects, size is not everything, but it does influence most of the things like cost and resource. So if we do not have an accurate prediction of size, it’s difficult to plan. A precise software to calculate the software cost estimation will be helpful to project managers to estimate their software size.

There are many factors, which lead to software projects fails. The major causes of software failure are poor planning and cost estimation. The initial cost and estimated schedule are not more frequently revised as more information available. Besides that, current practices in software cost estimation are being done manually and causes to project's failures. Such a major problem can be avoided if a software tool used to calculate the cost estimation of the projects to get an accurate result in estimating. An accurate and efficient cost estimation methodology for web-based application is very important for software development as it would assist the management team to estimate the cost. Furthermore, it will ensure the development of cost suits the planned budget and provides a fundamental motivation for the development of a web-based application project (Zulkefli, M., Zarinah, M.K., Habibah, A., Saadiah, Y, 2011). There have been various cost estimation models and methods that are being used in the software-development process. Function Points is one the example to calculate the estimation cost.

Function point is one of the most accepted and robust sizing techniques used in the software cost estimation process, function point, which formulate by Albrecht was established in the early of 1970 (M.A. Al-Hajri, A.A.A. Ghani, M.S. Sulaiman, M.H. Selamat. 2005). Function point has many advantages over the other cost estimation models like they are independent of programming language, product design or development style; it is a well-documented method and many more. In addition, function point estimation is achieved directly without the formalization of step by step analytical procedures. Besides that, research did by Graham C. Low and D. Ross Jeffery has proven that function point counts appear to be more consistent with measure software size.

1.2 Problem statement

Software cost estimation has a great impact on the software-development process. The success of the software project depends on factors such as time and cost. There have been researched conducted on this issue in Malaysia. However, the data used to have not been sourced locally. Therefore, the accuracy is questionable. Moreover, the factors can be dependent on local environment and needs specific to those in Malaysia. There is still much to be discovered and that is what this study is aimed at. According to a study made in Malaysia, they have come up with the statistic based on literatures that 52.7% projects were not able to be complete on time and over budget and 31.1% not fulfilled the scope (Iman.A, Ow.S.H.008.)

The causes of the project failure were mainly poor planning and estimation. Besides that, case studies on a group of50students were taken. 49undergraduates were from Faculty of Computer Science and Information Technology and an undergraduate from the department of Information science from University of Malaya, who took the course Project Management. The students were assigned a team project based on their preference. The students were divided into seven groups with seven to eight members comprising Malays, Chinese and Indians. All the projects were focused on the budget, schedule and quality. Based on the lecturer’s assessment on the students projects based on budget only one team managed to complete the project within the budget. The remaining six teams were over the budget. Besides that, analysis made on the cost estimation, two groups budgets were over the project cost, and four teams manage to make good estimation but failed to complete

within budget. Another research was also carried out based on estimation using function point and source of line code. The research was carried out based on two programs using specification prepared by an experienced professional analyst (G.C. Low And D.R.Jeffery. 1990). The 1st program used was Fixed Asset Master File Update, and the 2nd program used was Fixed Asset Depreciation Calculation and Reporting. The data collected, and the relationship was divided into three major data sets. The first data set was comprised of twenty-two function point experienced analyst counting a priori from the program specification. The second data set was comprised of two groups of function point naïve analysts where one group very experienced in analysis and the other group do not. The third data set was consisting of twelve analyst estimating source lines of code from the program specification. The result of their research on the comparison of the consistency of the function point and source lines of code estimates suggests that on an organizational basis source bank line of codes, estimates are no better than function point estimates when estimating from a program specification. The research concluded that function point counts appear to be a more consistent a a-priori measure of software size than source lines of codes. With the assistant of software estimation tool and a correct methodology used to calculate the estimation; such a problem can be avoided. By having software for the estimation calculation, the time taken to calculate for estimation can be reduced. Thus, the additional time left can be reallocated to focus on the difficult part on the software project phase like designing.

1.3 Objective

i. To analyze the consistency of the function point in measuring the software cost estimation

ii. To compare and find out the best way to practice the software cost estimation between function point and other method.

iii. To choose the best estimation method using the Analytical Hierarchy Process.

1.4 Scope

The study is carried out on the method of calculating the software cost estimation. The prototype focuses on the function point method. This prototype will be a standalone system. This project focuses on two modules of Ump- Automatic Sports Facilities Management System.

1.5 Thesis Organization.

Chapter one is mainly about the details of the case study. It contains introduction, problem statement, scope, and objective. Chapter two is about the literature review. A literature review is an account of what has been published on a topic accredited scholars and researchers. It is a part of the introduction to research report. The purpose is to convey the reader what knowledge and idea have been established on a topic and their strengths and weaknesses are. Chapter three is basically about the methodology used in this research paper. This chapter discusses about identifying the data collection instrument to be employed for the study, determining the sample or participants of the study, and analyses the data collected for the study. Chapter four discusses about the design of the system. Chapter five is about the implementation of the system followed by chapter six, which is about the result and discussion of the thesis. Finally, chapter seven is the conclusion of the thesis.

CHAPTER 2

LITERATURE REVIEW

2.1 Estimation System

Estimation is the process of finding an estimate, or approximation, which is a value that is usable for some purpose even if input data may be incomplete, uncertain, or unstable. Typically, estimation involves "using the value of a statistic derived from a sample to estimate the value of a corresponding population parameter."

Estimation is one of the most important parts in the software planning phase. The estimation involves estimating the size of the software, budget, time and resource. Errors or mistakes in estimation can attribute to many factors. The meaning of an estimation system in this context is estimating the size of the software. An accurate estimate of software size is an essential element in the calculation of estimated project costs and schedules. The fact that these estimates are required very early in the project (often while a contract bid is being prepared) makes size estimation a formidable task. If the estimation is inaccurate there will be major problems. If the under estimation was done, it can result in under-staffing and may result in an over worked and burnt-out team. The estimation process has sub tasks. As the size increases, the interdependency among various elements of the software grows rapidly. There are different methodologies for arriving at and expressing the size/complexity of the Software Program. Some of the popular ones are Function Points, Lines of Codes, and Cocomo.

In conclusion, estimation is extremely a vital process in a software planning phase. Thus, having a good estimation system to estimate the size of the system would be an advantage. In addition, it will make sure that we would keep us on the track to complete the system on time and with the entire requirement are fulfilled.

2.2 Existing Application and its Problems

2.2.1 Lines of Code

Lines of code often referred as Source Lines of Code, SLOC or LOC. Lines of code are a formal method to measure the size by counting number of lines of Code. This metric was very popular primarily because of its use simple and easy. Lines of code measure the number of source instruction used to solve a problem. While counting the number of source instructions, lines used for commenting and blank lines are ignored. Although using lines of code is simple but there are many disadvantages in using it.

Usually estimation will be done at the beginning of the project but by using lines of code estimation at the beginning of the project will be very tricky. In order to make the estimation easy, the project manager will divide the project managers divide the problem into modules, and each module into sub modules, and so on until the sizes of the different leaf-level modules can be approximately predicted. The next disadvantage is some programmers may create a lengthy code structure to solve a problem as they do not make effective use of the available instruction set. At the same time, skilled programmers can code a simple and effective code for the same problem. Thus, a poorly written code cannot be a good metric for estimation purpose.

Lines of code only focus on the coding part and ignore other important parts such as designing and implementation. Coding is only a small part in a software development system. Besides that, LOC also have problem with its language dependence. LOC only allow usage of one language in the estimation process unlike function point that does not depend on what language used. It is not possible to directly compare projects developed by using different languages. For example, the time per line for a high-level

language may be greater than for a lower-level language. There is no way to accommodate the fact that fewer lines of code may be required for a higher-level language to provide the same function. Moreover, LOC is lack of a universally accepted definition for exactly what a line of code really is. Jones (1986) identified 11 major variations of line counting methods. Since few authors state the line-counting rules they used, much of the literature has an “uncertainty of perhaps 500% attributable to line counting variations.” The variations make it very difficult to compare studies using lines of code as a measure of software size (Matson.J.E, Barrett.E.B., Mellichamp.J.M., 1994). In conclusion, with so many disadvantages, using LOC for estimating a software system is not a good idea.

2.2.2 Constructive Cost model (COCOMO)

Constructive Cost Model (COCOMO) is an algorithmic software cost estimation model. It was developed by Barry W. Boehm. The model is a combination of statistical figures, mathematical equations and expert judgments. Furthermore, COCOMO is an open model, so all the details such as the underlying cost estimation equations, Every assumption made in the model, Every definition, and The costs included in an estimate are explicitly stated. This model also have been widely use in the companies. COCOMO is composed into three levels of the models which is basic, intermediate and detailed. The basic COCOMO'81 model is a single-valued, static model that computes software development effort (and cost) as a function of program size expressed in estimated thousand delivered source instructions.

The intermediate COCOMO'81 model computes software development effort as a function of program size and a set of fifteen "cost drivers" that include subjective assessments of product, hardware, personnel, and project attributes. Some of the COCOMO’s limitations are primarily COCOMO represents development from planning to implementation. It doesn’t consider maintenance, rework, porting and integration, and reuse. COCOMO model ignores requirements and all documentation. Function point method measures the developed system by point counts that can determined relatively early in the development process. It measures software project size by studying external features of the projects (Gao.X, Lo.Bruce, .1995). So it avoids the difficulty of the COCOMO method which means in COCOMO does not support some newer programming environment like Authorware, make counting of LOC difficult as the definition of a line is

less clear-cut. Besides that, it ignores customer skills, cooperation, knowledge and other parameters. It oversimplifies the impact of safety/security aspects. It ignores hardware issues. It ignores personnel turnover levels. It is dependent on the amount of time spent in each phase.

2.3 Function point

The Function Point methodology was developed by Allan Albrecht at IBM in 1979. This methodology is based on the belief that the size of a software project can be estimated during the requirements analysis. Function Point Analysis (FPA), or the method of sizing software in terms of its function and expressed in Function Points is now very widely used (Vicker.P.). It takes into account the inputs and outputs of the system. Function points can be determined from the requirement specifications, design specification, source listing or live system. Function point focuses on “functionality” or “utility” rather than counting LOC (Gao.X, Lo.Bruce, .1995). Since function point measure functionality, they should be independent of technology and language used for the software implementation (Low,G.C. And Jeffery,D.R. 1990). It helps to estimate software effort more accurately without considering the languages or developing environment you choose (Zheng.Y, Wang.B, Zheng.Y, Shi.L. 2009).In addition, the ability of function point could help in effort, schedule, and defect estimation and aids in setting project scope. Function point does not counts the lines like how LOC does instead it counts the number of externals that make up the system. The function point approach has features that overcome the major problems with using lines of code as a measure of system size.

First, function points are independent of the language, tools, or methodologies used for implementation; i.e., they do not take into consideration programming languages, data base management systems, processing hardware, or any other data processing technology (Matson.J.E, Barrett.E.B., Mellichamp.J.M., 1994). Function points are based on the system user’s external view of the system, nontechnical users of the software system have a better understanding of what function points are measuring (Matson.J.E, Barrett.E.B., Mellichamp.J.M., 1994). By using function point, its make us easy for comparing project productivity and organization besides help manager

better understand and articulate to client the impact of change request and enhancements. Function points also allow the manager to make informed decision more objectively with the limited time and budget.

A case study done by few researches have proved that relationship between software effort and function point count can be assumed as a linear regression model y = a + bx where *a* is the slope of the line (gradient) and *b* is the y intercept. From the scatter diagram which was shown in the figure 1,the relationship between the variable and the dependent variable tends to be a straight line with highly positive correlation. The case study also concluded that though it is difficult to figure out the count of function point, the linear function will greatly simplify the process of software estimation, and then help the manager to have an accurate software effort measurement (Zheng.Y, Wang.B, Zheng.Y, Shi.L. 2009).

Another case study which was carried out in 1994 has also come out with a conclusion which is Function point counts appear to be a more consistent a priori measure of software size than source lines of code (Low,G.C. And Jeffery,D.R. 1990).

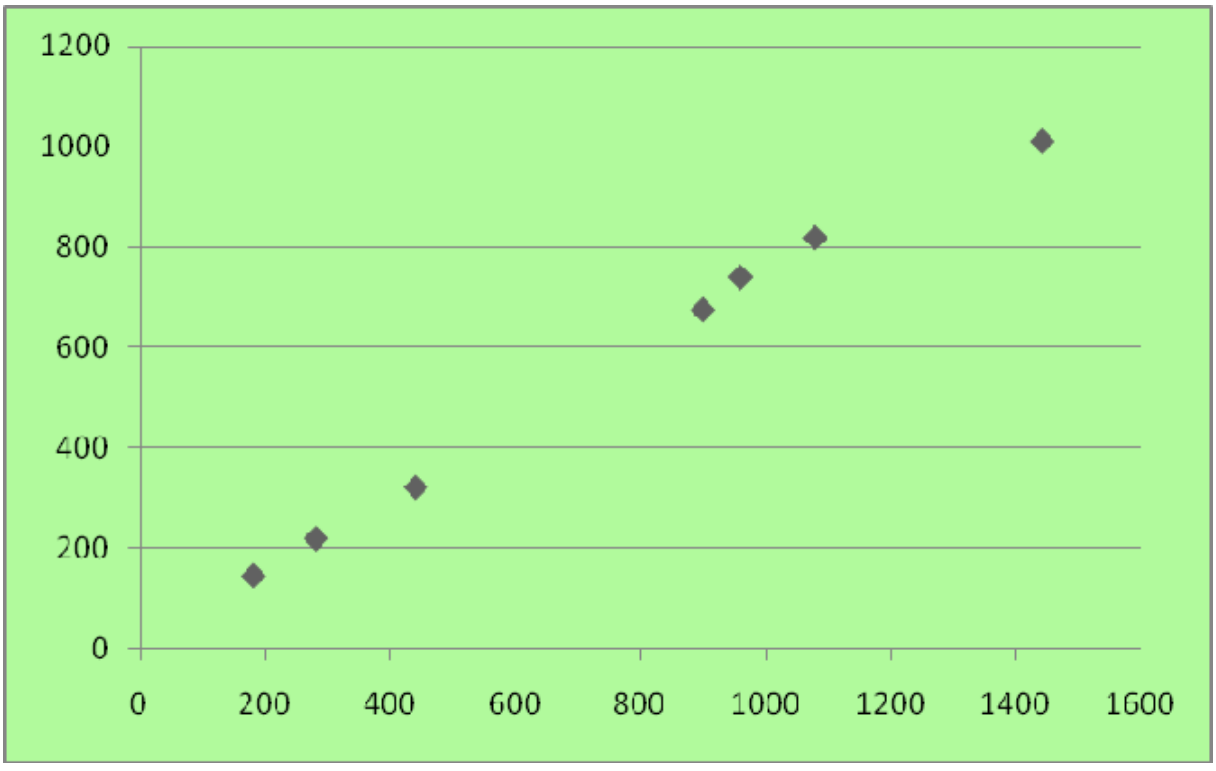

Figure 2.1 Relation between function point and software effort

Example of a linear regression model. The vertical axis is the x-axis that represents the software effort and the horizontal axis the y-axis that represents the Function point.

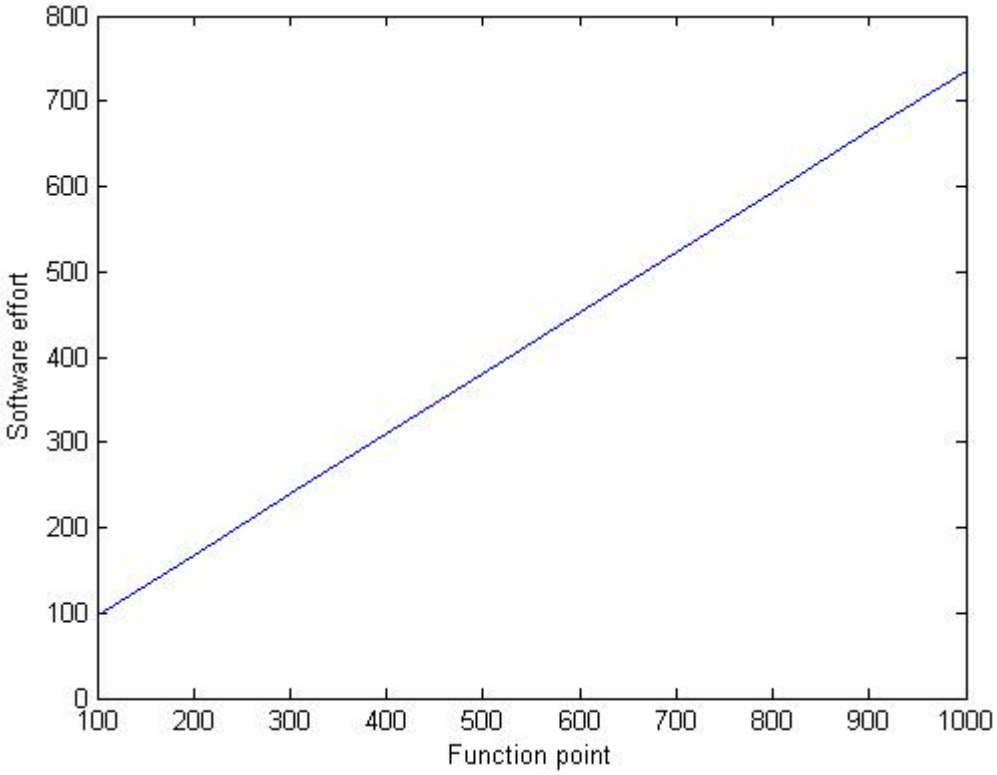

Figure 2.2 Linear relation between function point and software effort

2.4 Applied Function Point application in the Industry.

The function point methodology is being successfully applied by innumerable organizations world-wide to measure software size for existing applications, enhancements to those applications, and new development projects. Function Points Analysis (FPA) has become a world standard over the years. The mission of International Function Point Users Group (IFPUG) is to be a recognized leader in promoting and encouraging the effective management of application software development and maintenance activities through the use of Function Point Analysis and other software measurement techniques. IFPUG endorses FPA as its standard methodology for software sizing (Dekkers.T). Carol Dekkers from Quality Plus Technologies and Mauricio Aguiar from Caixa Economica Federal has stated that trough their experience shows that FPA is often more effective than peer or user walkthroughs in identifying the full set of functional user requirements and uncovering potential defects. In fact, benefits gained by applying FPA to functional user requirements can be more valuable than the mere function point size of the software (Carol A.D., Aguiar.M).

There are many reasons why most of the industries choose Function Point as a tool for estimating. The outcome of a Function Point count provides the metric ‘unit of software delivered and can be used to assist in the management and control of software development, customization or major enhancements from early project planning phases through to the ongoing support of the application. In addition, the software size facilitates the creation of more accurate estimates of project resources and delivery dates and facilitates project tracking to monitor any unforeseen increases in scope. Industry figures available from International Software Benchmarking Standards Groups (ISBSG) *Repository* for projects measured with IFPUG function points indicates that complete applications tend to have consistent and predictable ratios of each of the function types.

The research conducted by the Total Metrics Pty. Ltd shows that industry figures show that the risk of project failure rapidly increases with project size. Projects less

than 3500 function points have a risk of failure of less than 20% in comparison with projects over 5000 function points which have a probability of cancellation close to 40%. This level of risk5 is unacceptable for most organizations. Data within the ISBSG Repository Release 6 supports the premise that smaller projects are successful. Over 65% of the projects in the repository are less than 500 function points and 93% of the projects are less than 2000 function points. Thus, they concluded that Industry experience suggests that the best managed projects which deliver quality software on time and within budget tend to less than 700 function points and up to 1500 function points.

While in Malaysia, many projects were developed but only very little percentage of projects that have succeed while other succeed with challenges such as over run budget, time overrun, and impaired functionality. There are many factors which can lead to such condition but the main factor would be choosing the wrong method to estimate the software size and budget. A research based on software cost estimation practices in Malaysia was conducted by Zulkefli, M., Zarinah, M.K., Habibah, A., and Saadiah, Y in the year 2012. The research has come out with a random survey in order to get an overview of current practice in cos testimation process among project managers and web developers in Klang Valley. The research have concluded that Both project managers and web developers agreed that Expert Judgment, Price-to-Win and Algorithmic Model methods produce most accurate result in cost estimation process and at the same time the result is incongruent to the method preferable used by the project manager and web developer. The method Parkinson-Ian shows the most accurate method to count the cost estimation. The paper conclude that in order to get accurate cost estimation result a good estimation process with the proper selection of cost estimation technique, correct size measure, person experiences, and familiarity of software developed.

We do have SIRIM and International Organization for Standardization (ISO) in Malaysia. SIRIM Berhad is a wholly-owned company of the Malaysian Government under the Ministry of Finance Incorporated. SIRIM is recognized the world over as a global research and standards development organization. SIRIM focus on discovering and developing new technologies to help businesses compete better through quality and innovation. Besides that, SIRIM also continuously reinventing the way we do things and ensuring that they remain market-driven, flexible, cost-effective, and responsive to our

clients. ISO organization is responsible for developing standards for products and services that identify a need for standardization. The ISO is usually contacted by a sector of an industry or stakeholders in a product and asked to develop a standard, such as those created for manufacturing. ISO helps governments around the world create environmental, health and safety policies. The ISO helps test products during the standard-setting process. Many international trade agreements also incorporate ISO standards. ISO standards allow consumers to buy and use products safely. The use of ISO branding allows products to be produced safely and efficiently.